Chapter 3

Digital Signature

COMPARISON

Differences between conventional signatures and digital signatures.

1. **Inclusion**
2. **Verification Method**
3. **Relationship**
4. **Duplicity**

1. Inclusion

- ***A conventional signature:***
- ***included in the document; it is part of the document.***
- ***Digital signature:***
- ***The signature as a separate document.***
- ***Sender send two document:-***
  1. ***Message***
  2. ***Digital signature***

2. Verification Method

- ***conventional signature:***
- ***Receiver compares the signature on the document with the signature on file.***
- ***digital signature:***
- ***Copy is not stored anywhere.***
- ***receiver receives the message and the signature and apply a verification technique to the combination of the message and the signature to verify the authenticity.***

3. Relationship

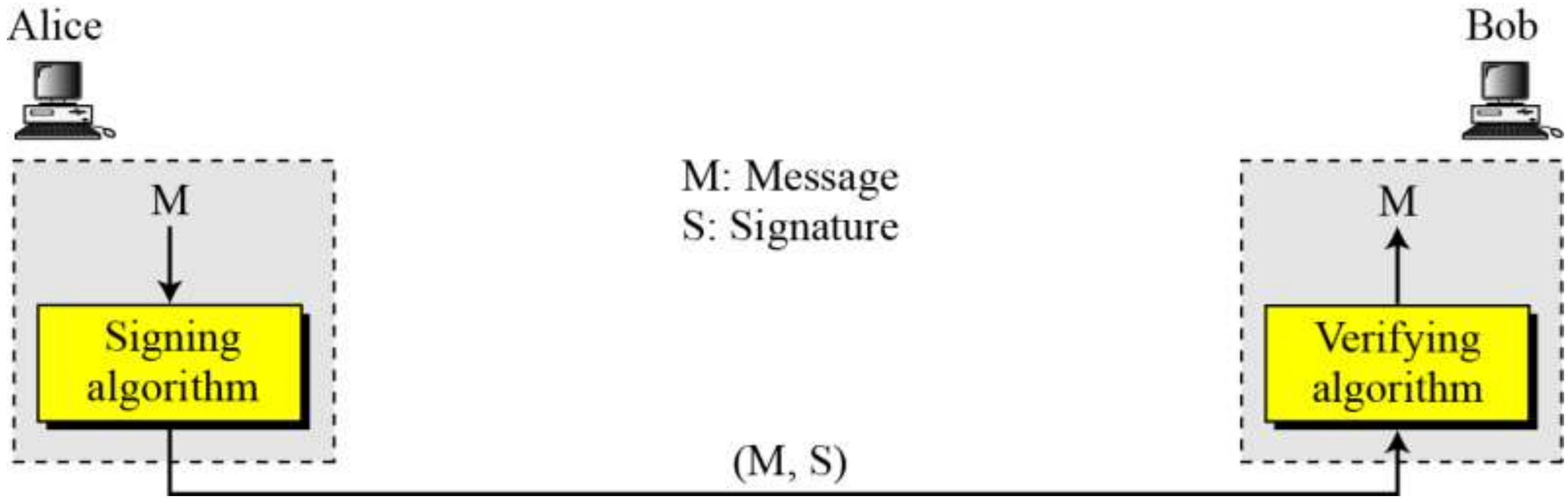
- ***Conventional signature:***
- ***one-to-many relationship between a signature and documents.***
- ***one person uses same signature on many documents.***
- ***digital signature:***
- ***one-to-one relationship between a signature and a message.***
- ***Most of the time each message needs new signature.***

4. Duplicity

- ***conventional signature:***
- ***a copy of the signed document can be distinguished from the original one on file.***
- ***digital signature:***
- ***there is no such distinction unless there is timestamp on the document.***
- ***<u>For example:</u> Alice sends a document instructing Bob to pay Eve. If Eve intercepts a document and signature, she can replay it later to get money again from Bob.***

PROCESS

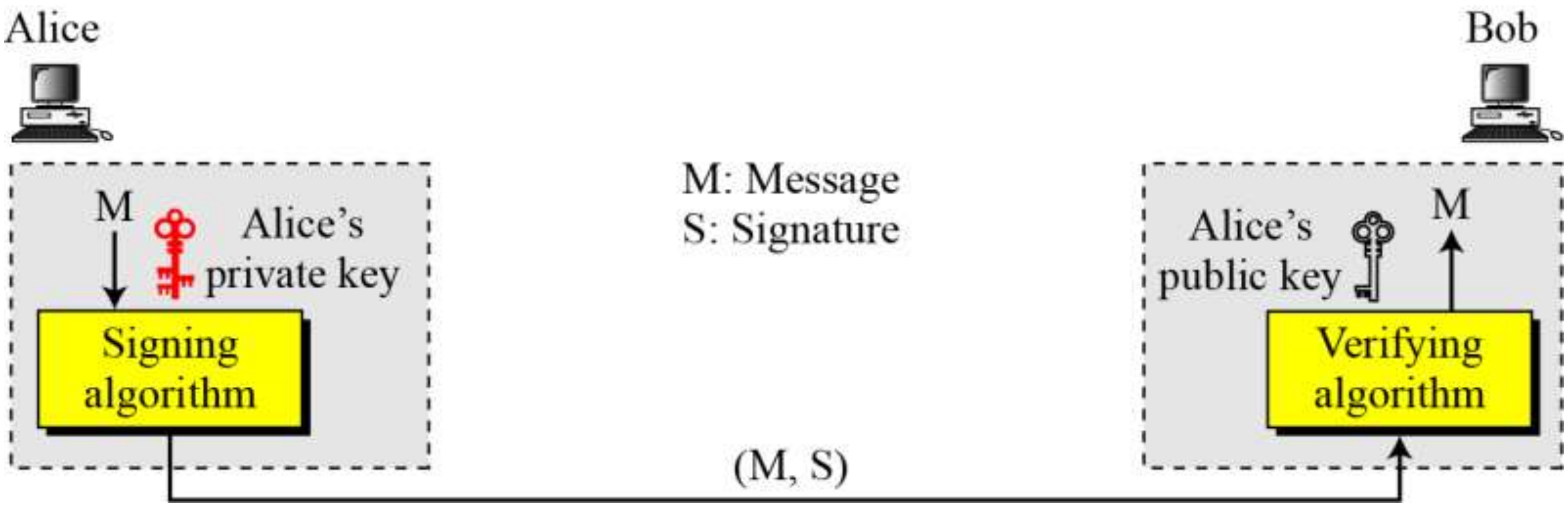
Digital signature process



- ***Sender uses a signing algorithm to sign the message.***
- ***Message and the signature are sent to the receiver.***
- ***Receiver receives both and the signature and applies the verifying algorithm to the combination.***
- ***If the result is true, the message is accepted; otherwise, it is rejected.***

Need for Keys

Adding key to the digital signature process



A digital signature needs a public-key system. The signer signs with her private key; the verifier verifies with the signer's public key.

SERVICES

Digital Signature provides following services:

1. **Message Authentication**
2. **Message Integrity**
3. **Nonrepudiation**
4. **Confidentiality**

1. *Message Authentication*

- ***Data origin authentication***
- ***Bob can verify that message has come from Alice because Alice's public key is used for verification.***
- ***Alice's public can not verify signature signed by any other's private key.***

A digital signature provides message authentication.

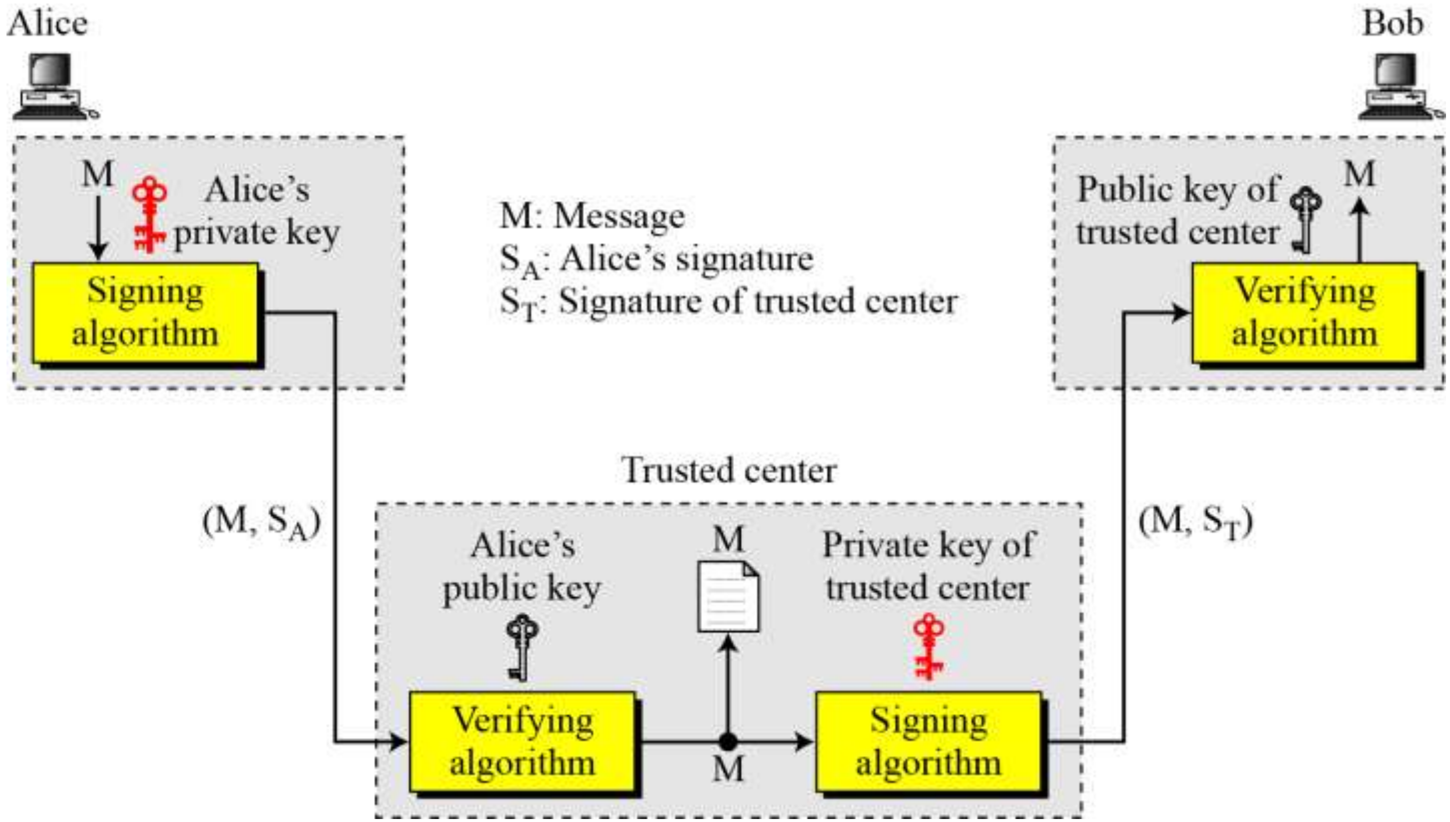
2. Message Integrity

We cannot get the same signature if the message is changed.

A digital signature provides message integrity.

3. Nonrepudiation

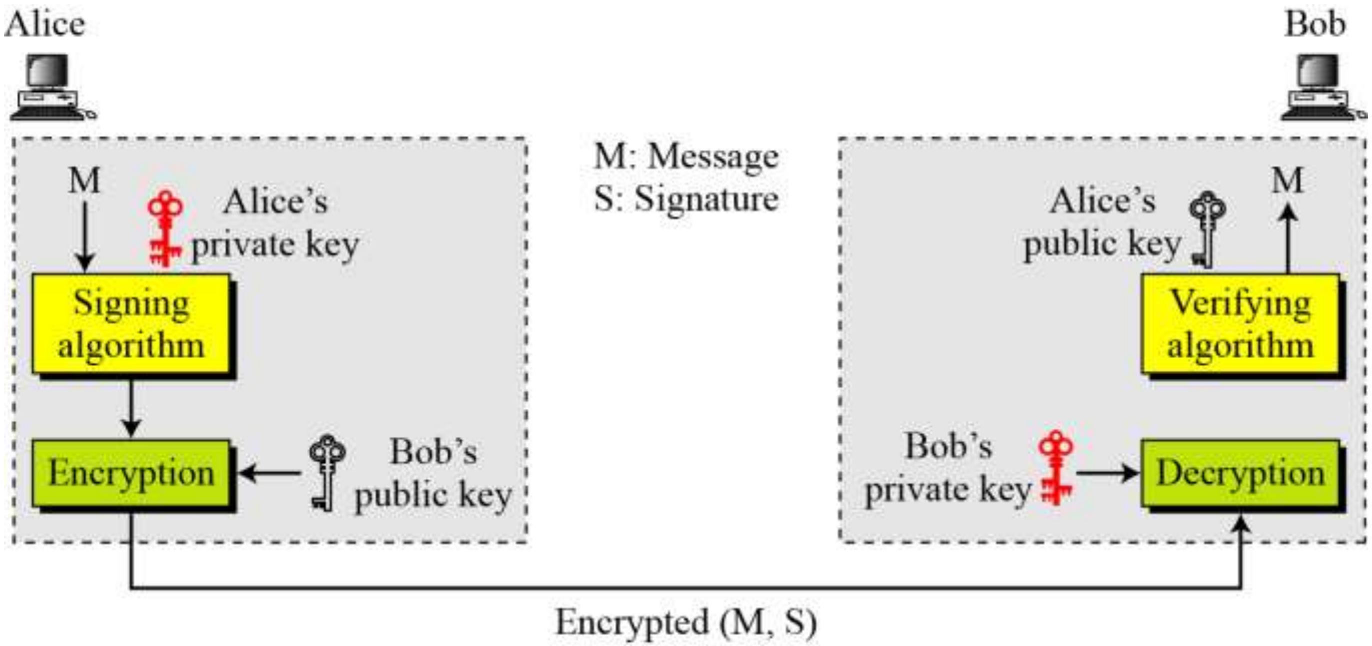
Using a trusted center for nonrepudiation



Nonrepudiation can be provided using a trusted party.

4. Confidentiality

Adding confidentiality to a digital signature scheme



A digital signature does not provide privacy. If there is a need for privacy, another layer of encryption/decryption must be applied.

ATTACKS ON DIGITAL SIGNATURE

- **Attack Types**
  - ***Key-Only Attack***
  - ***Known-Message Attack***
  - ***Chosen-Message Attack***

- **Forgery Types**
  - ***Existential Forgery***
  - ***Selective Forgery***

Attack Types

Key-Only Attack

- Attacker knows only public key released by Alice.
- To forge message, attacker needs to create Alice's signature.

Known-Message Attack

- Attacker has some documents previously signed by Alice.
- Attacker tries to create another message and forge

Attack Types

Chosen-Message Attack

- ***Attacker somehow makes Alice sign one or more messages for him.***
- ***Attacker now has chosen-message /signature pair.***
- ***Attacker later creates another message, and forge Alice's signature on it.***

Forgery Types

Existential Forgery

- ***Document is forged, but the content is randomly calculated; unfortunately the document is syntactically or semantically unintelligible.***
- ***Attacker can not get benefit from it.***

Forgery Types

Selective Forgery

- ***Attacker may be able to forge** Alice's **signature on a message with the content selectively chosen by him.***
- ***This is beneficial to attacker but may be very detrimental to Alice.***

The probability of such forgery is low , but not negligible.

DIGITAL SIGNATURE SCHEMES

- A digital signature scheme is a mathematical scheme for demonstrating the authenticity of a digital message or document.
- A digital signature scheme typically consists of three algorithms:
  - A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.
  - A signing algorithm that, given a message and a private key, produces a signature.
  - A signature verifying algorithm that, given a message, public key and a signature, either accepts or rejects the message's claim to authenticity.

DIGITAL SIGNATURE SCHEMES

1. **RSA Digital Signature Scheme**
2. **ElGamal Digital Signature Scheme**
3. **Schnorr Digital Signature Scheme**
4. **Digital Signature Standard (DSS)**

1. RSA Digital Signature Scheme

Key Generation

Key generation is exactly the same as key generation in the RSA

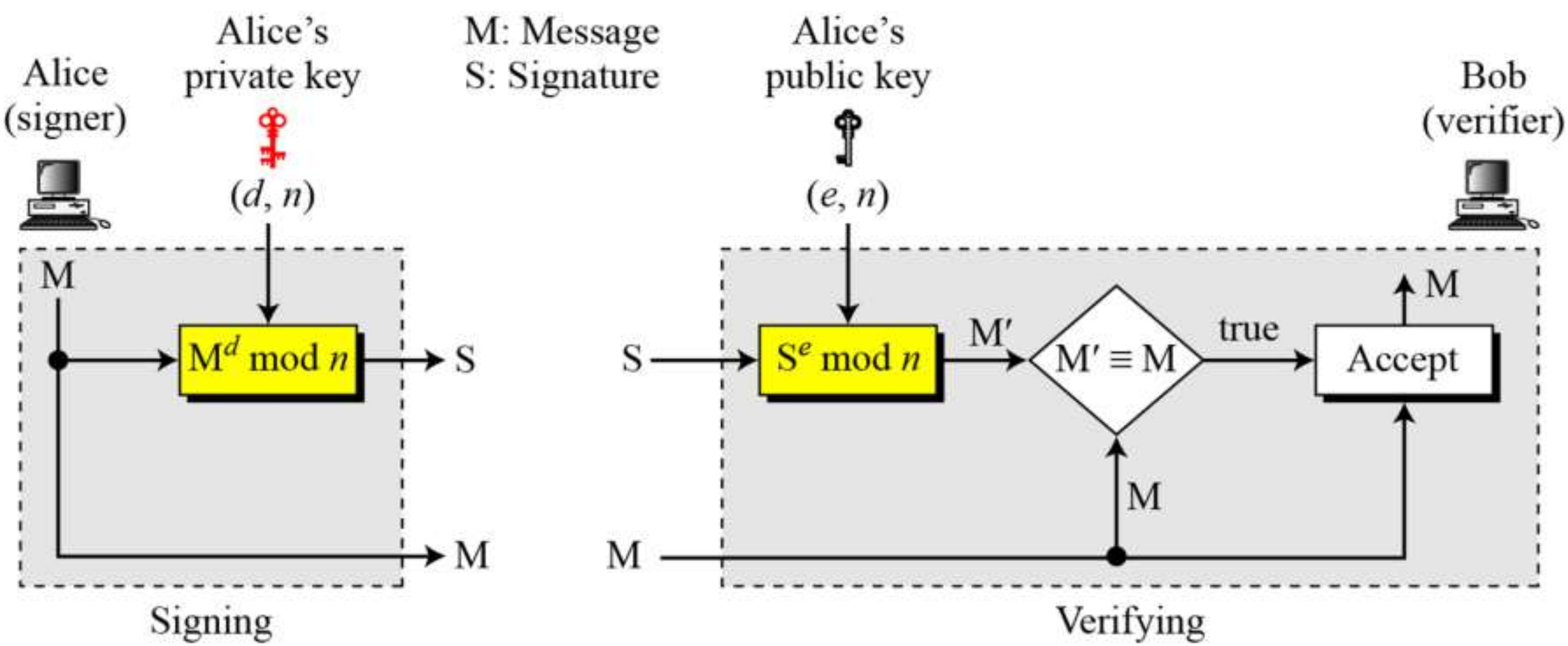
- *Alice chooses 2 prime numbers p and q and calculates $n = p*q$*
- *Alice calculates $\Phi(n) = (p-1)(q-1)$*
- *She then chooses e, the public exponent, and calculates d, the private exponent such that $e*d = 1 \bmod \Phi(n)$*
- *Alice keeps d and publicly announces e and n*

In the RSA digital signature scheme, *d* is private; *e* and *n* are public.

1. RSA Digital Signature Scheme

Signing and Verifying

RSA digital signature scheme



1. RSA Digital Signature Scheme

Example

- **Alice chooses *p* = 823 and *q* = 953, and calculates *n* = 784319.**
- **φ(n)=782544.**
- **Suppose Alice chooses *e* = 313 and calculates *d* = 160009.**
- **Suppose Alice sends a message M = 19070 to Bob.**
- **Alice uses private exponent, *d*=160009, to sign the message:**

$$\text{M: } 19070 \quad \rightarrow \quad S = (19070^{160009}) \bmod 784319 = 210625 \bmod 784319$$

Alice sends the message and the signature to Bob. Bob receives the message and the signature. He calculates

$$M' = 210625^{313} \bmod 784319 = 19070 \bmod 784319 \quad \rightarrow \quad M \equiv M' \bmod n$$

Bob accepts the message because he has verified Alice's signature.

2. ElGamal Digital Signature Scheme

Key Generation

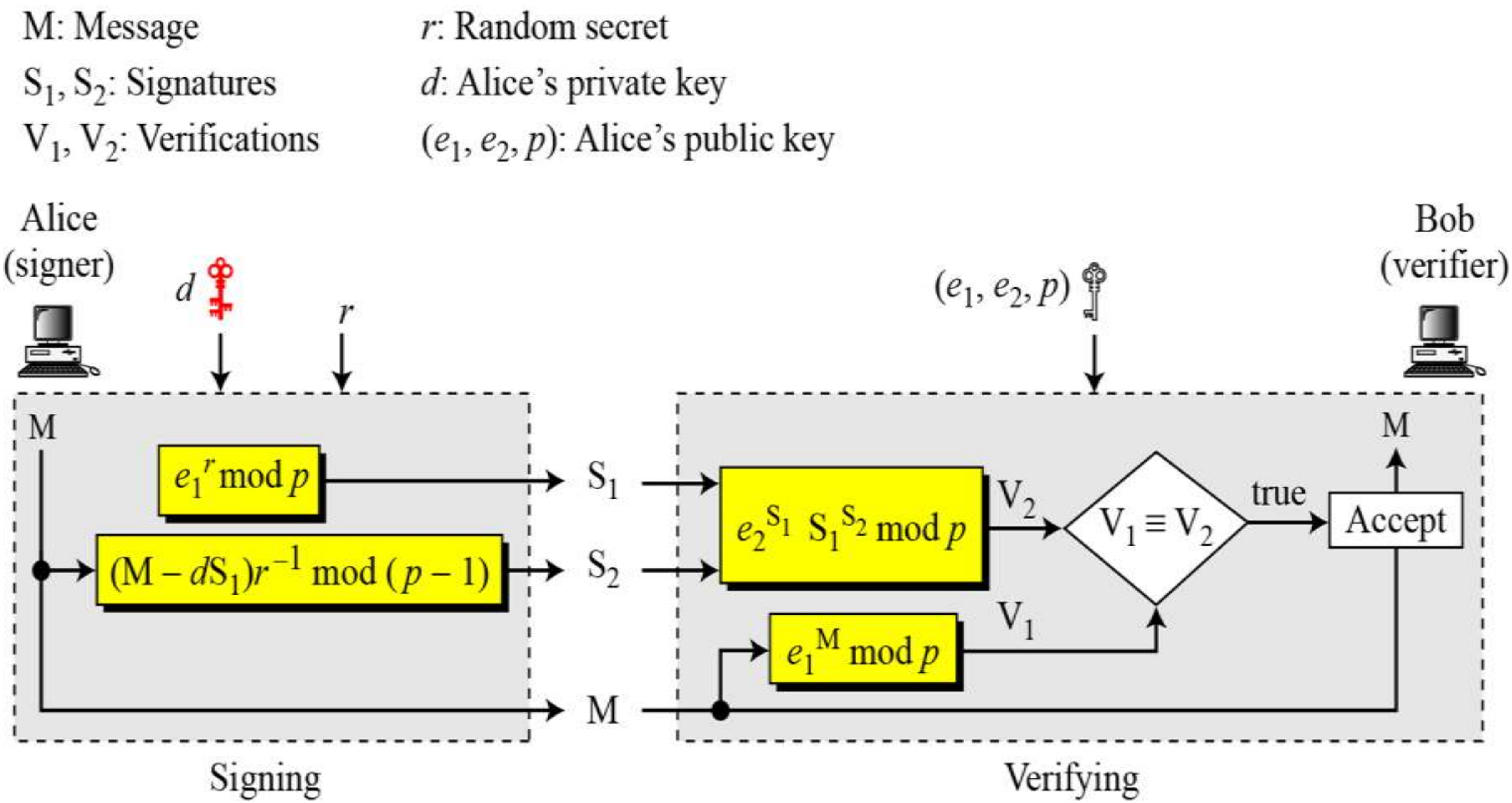
- ***p* is large enough prime number.**
- ***Let e1 is primitive root in Zp****
- ***Choose randomly a secret key e_1 with $1 < x < p - 1$.***
- ***Compute $e_2 = e_1^{d} \bmod p$.***
- ***The public key is (p, e_1, e_2).***
- ***The secret key is d, less than p-1.***

In ElGamal digital signature scheme, (e_1, e_2, p) is Alice's public key; *d* is her private key.

2. ElGamal Digital Signature Scheme

Verifying and Signing

ElGamal digital signature scheme



2. ElGamal Digital Signature Scheme

Example

Here is a trivial example. Alice chooses p = 3119, e_1 = 2, d = 127 and calculates $e_2 = 2^{127}$ mod 3119 = 1702. She also chooses r to be 307. She announces e*1*, e*2*, and p publicly; she keeps d secret. The following shows how Alice can sign a message.

$$M = 320$$

$$S_1 = e_1{}^r = 2^{307} = 2083 \bmod 3119$$

$$S_2 = (M - d \times S_1) \times r^{-1} = (320 - 127 \times 2083) \times 307^{-1} = 2105 \bmod 3118$$

Alice sends M, S_1, and S_2 to Bob. Bob uses the public key to calculate V_1 and V_2.

$$V_1 = e_1{}^M = 2^{320} = 3006 \bmod 3119$$

$$V_2 = d^{\,S_1} \times S_1{}^{S_2} = 1702^{2083} \times 2083^{2105} = 3006 \bmod 3119$$

2. ElGamal Digital Signature Scheme

Example

Now imagine that Alice wants to send another message, M = 3000, to Ted. She chooses a new *r*, 107. Alice sends M, S_1, and S_2 to Ted. Ted uses the public keys to calculate V_1 and V_2.

$$M = 3000$$

$$S_1 = e_1{}^r = 2^{107} = 2732 \bmod 3119$$

$$S_2 = (M - d \times S_1)\, r^{-1} = (3000 - 127 \times 2732) \times 107^{-1} = 2526 \bmod 3118$$

$$V_1 = e_1{}^M = 2^{3000} = 704 \bmod 3119$$

$$V_2 = d^{S_1} \times S_1{}^{S_2} = 1702^{2732} \times 2732^{2526} = 704 \bmod 3119$$

3. Schnorr Digital Signature Scheme

Key Generation

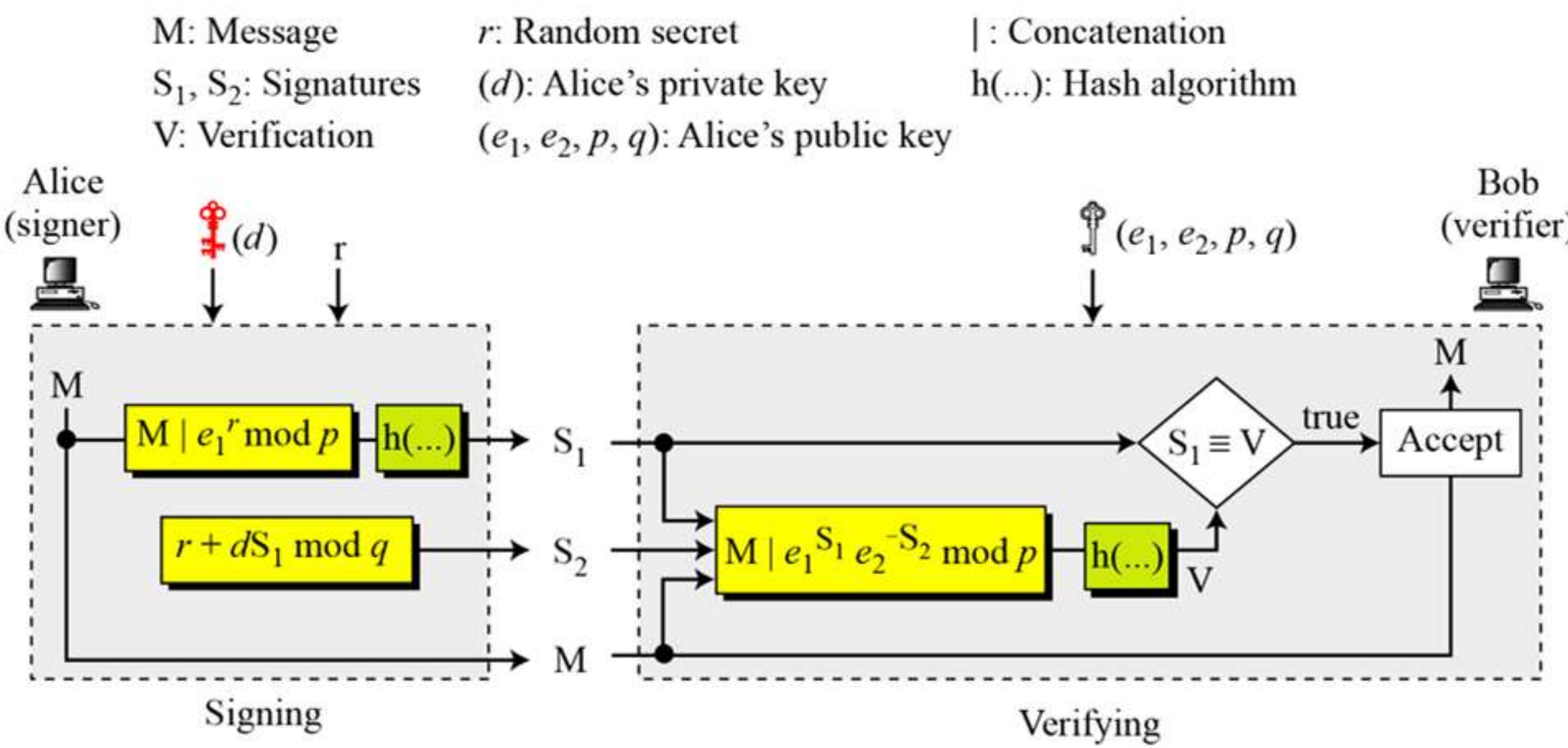
1) *Alice selects a prime p, which is usually 1024 bits in length.*
2) *Alice selects another prime q.* Such that (p-1)=0 mod q
3) *Alice chooses e_1 to be the qth root of 1 modulo p.* Such that, $e1=e_0^{(p-1)/q}$ mod p, e0 is first primitive root of p
4) *Alice chooses an integer, d, as her private key.*
5) *Alice calculates $e_2 = e_1^d \bmod p$.*
6) *Alice's public key is (e_1, e_2, p, q); her private key is (d).*

Note

In the Schnorr digital signature scheme, Alice's public key is (e_1, e_2, p, q); her private key (d).

3. Schnorr Digital Signature Scheme

Signing and Verifying



3. Schnorr Digital Signature Scheme

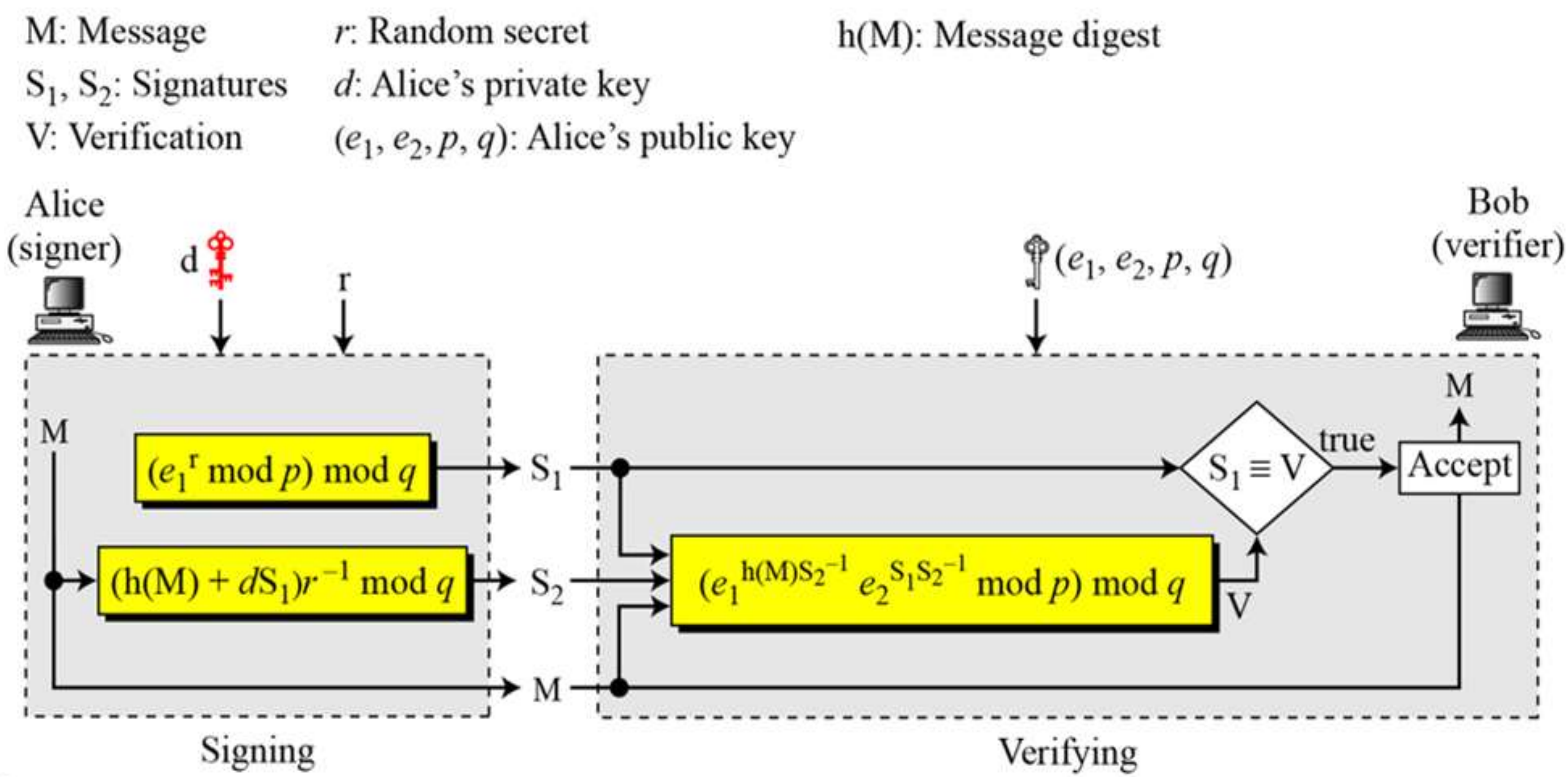
Signing

1. *Alice chooses a random number r.*
2. *Alice calculates $S_1 = h(M|e_1^r \bmod p)$.*
3. *Alice calculates $S_2 = r + d \times S_1 \bmod q$.*
4. *Alice sends M, S_1, and S_2.*

Verifying Message

1. *Bob calculates $V = h\ (M \mid e_1^{S2}\ e_2^{-S1} \bmod p)$.*
2. *If S_1 is congruent to V modulo p, the message is accepted;*

4. Digital Signature Standard



4. Digital Signature Standard

Key Generation

Before signing a message to any entity, Alice needs to generate keys and announce the public ones to the public.

1. Alice chooses a prime p, between 512 and 1024 bits in length. The number of bits in p must be a multiple of 64.
2. Alice chooses a 160-bit prime q in such a way that q divides $(p - 1)$.
3. Alice uses two multiplication-groups $<\mathbf{Z}_p^*, \times>$ and $<\mathbf{Z}_q^*, \times>$; the second is a subgroup of the first.
4. Alice creates e_1 to be the qth root of 1 modulo p ($e_1^p = 1 \bmod p$). To do so, Alice chooses a primitive element in $\mathbf{Z}_p$, e_0, and calculates $e_1 = e_0^{(p-1)/q} \bmod p$.
5. Alice chooses d as the private key and calculates $e_2 = e_1^d$.
6. Alice's public key is (e_1, e_2, p, q); her private key is (d).

4. Digital Signature Standard

Signing The following shows the steps to sign the message:

1. Alice chooses a random number r ($1 \le r \le q$). Note that although public and private keys can be chosen once and used to sign many messages, Alice needs to select a new r each time she needs to sign a new message.
2. Alice calculates the first signature $S_1 = (e_1^r \bmod p) \bmod q$. Note that the value of the first signature does not depend on M, the message.
3. Alice creates a digest of message h(M).
4. Alice calculates the second signature $S_2 = (h(M) + d\,S_1)r^{-1} \bmod q$. Note that the calculation of S_2 is done in modulo q arithmetic.
5. Alice sends M, S_1, and S_2 to Bob.

4. Digital Signature Standard

Verifying Following are the steps used to verify the message when M, S_1, and S_2 are received:

1. Bob checks to see if $0 < S_1 < q$.
2. Bob checks to see if $0 < S_2 < q$.
3. Bob calculates a digest of M using the same hash algorithm used by Alice.
4. Bob calculates $V = [(e_1^{h(M)S_2^{-1}} e_2^{S_1S_2^{-1}}) \bmod p] \bmod q$.
5. If S_1 is congruent to V, the message is accepted; otherwise, it is rejected.

4. Digital Signature Standard

DSS Versus RSA

Computation of DSS signatures is faster than computation of RSA signatures when using the same p.

DSS Versus ElGamal

DSS signatures are smaller than ElGamal signatures because q is smaller than p.